RADC-TR-88-324, Vol VI (of nine)
Interim Report
March 1989

NORTHEAST ARTIFICIAL INTELLIGENCE CONSORTIUM ANNUAL REPORT 1987 Artificial Intelligence Applications to Speech Recognition

Syracuse University

H.E. Rhody and J.A. Biles

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, NY 13441-5700

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS N/A
2a. SECURITY CLASSIFICATION AUTHORITY N/A	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) N/A	5. MONITORING ORGANIZATION REPORT NUMBER(S) RADC-TR-88-324, Vol VI (of nine)
6a. NAME OF PERFORMING ORGANIZATION Northeast Artificial Intelligence Consortium (NAIC) 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Rome Air Development Center (IRAA)
6c. ADDRESS (City, State, and ZIP Code) 409 Link Hall Syracuse University Syracuse NY 13244-1240	7b. ADDRESS (City, State, and ZIP Code) Griffiss AFB NY 13441-5700
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Rome Air Development Center 8b. OFFICE SYMBOL (If applicable) COES	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F30602-85-C-0008
8c. ADDRESS (City, State, and ZIP Code) Griffiss AFB NY 13441-5700	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 62702F; PROJECT NO. 5581; TASK NO. 27; WORK UNIT ACCESSION NO. 13
11. TITLE (Include Security Classification) NORTHEAST ARTIFICIAL INTELLIGENCE CONSORTIUM ANNUAL REPORT 1987 Artificial Intelligence Applications to Speech Recognition	
12. PERSONAL AUTHOR(S) H. E. Rhody, J. A. Biles Cont'd → Rhody's	
13a. TYPE OF REPORT Interim 13b. TIME COVERED FROM Dec 86 TO Dec 87	14. DATE OF REPORT (Year, Month, Day) March 1989 15. PAGE COUNT 44
16. SUPPLEMENTARY NOTATION This effort was performed as a subcontract by RIT Research Corporation to Syracuse University, Office of Sponsored Programs.	

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	
12	05		Artificial Intelligence, Speech Recognition, Expert Systems, Signal Processing, Phoneme Classification, Knowledge-based Systems Phonetics, Parsers, (TD)
12	07		

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The Northeast Artificial Intelligence Consortium (NAIC) was created by the Air Force Systems Command, Rome Air Development Center, and the Office of Scientific Research. Its purpose is to conduct pertinent research in artificial intelligence and to perform activities ancillary to this research. This report describes progress that has been made in the third year of the existence of the NAIC on the technical research tasks undertaken at the member universities. The topics covered in general are: versatile expert system for equipment maintenance, distributed AI for communications system control, automatic photo interpretation, time-oriented problem solving, speech understanding systems, knowledge base maintenance, hardware architectures for very large systems, knowledge-based reasoning and planning, and a knowledge acquisition, assistance, and explanation system. The specific topic for this volume is the design and implementation of a knowledge-based system to read speech spectrograms.

cont'd 9, 6-2

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL JOHN G. PARKER	22b. TELEPHONE (Include Area Code) (315) 330-4024 22c. OFFICE SYMBOL RADC/IRAA

DD Form 1473, JUN 86 Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

6 ARTIFICIAL INTELLIGENCE APPLICATIONS TO SPEECH RECOGNITION

Report submitted by:

Harvey E. Rhody
James Hillenbrand
John A. Biles

Rochester Institute of Technology
75 Highpower Road
Rochester, New York 14623

TABLE OF CONTENTS

6.1 Executive Summary

The primary goal the RIT NAIC project is the development of techniques that can be applied to the most difficult type of speech understanding system: the speaker independent, continuous speech, large vocabulary system. The problems posed by this type of system require an approach that is fundamentally different from the kinds of low level acoustically based engineering approaches that have traditionally been applied to speech recognition problems.

The major goals of our speech understanding effort are the development of techniques to: (1) derive an intermediate phonetic representation of spoken utterances from any speaker, (2) measure the quality of the match between errorful phonetic representations of input utterances and phonetically based lexical entries, (3) efficiently search large lexicons, and (4) incorporate syntax and domain knowledge in differentiating between plausible and implausible parsings of phonetic strings.

Our research on automatic phonetic analysis has proceeded along three related lines: (1) the development of low level feature extraction techniques that are designed to preserve information that is most closely associated with phonetic content, (2) the development of a feature based, speaker independent phonetic space, and (3) the development of learning based techniques for classifying featural representations of speech signals into phonetic categories. An example of our work in the first area is the development of a formant tracker that makes use of statistically based machine learning principles. An example of our work in the second area is the development of a relatively simple feature space that can be used to classify vowels produced by a diverse group of speakers based on formant frequency patterns. An example of our work in the third area is a tree search technique for classifying segments of spoken utterances into broad phonetic categories.

The goal of our work in phonetic string analysis is to develop methods to derive word sequences from an analysis of undifferentiated (i.e., without word boundaries) phonetic strings. This type of analysis is complicated not only by the presence of phonetic transcription errors, but also by the possibility that any segment in the input string may represent three different types of errors (substitution, omission or insertion). A project that is just underway uses a dynamic programming technique that provides a uniform way to address all three types of errors that can occur. A project that is slated to begin early in 1988 will extend this work by incorporating syntax and domain knowledge in the string parser.

6.2 System Architecture

We continue to view the system's architecture in two dimensions: a virtual software dimension and a physical hardware dimension. Most of the work done this year on the system architecture has been in the hardware dimension, but we also are filling in more details at the word and sentence levels of the software architecture, as the focus of recent theses has been at these levels.

Extensive developments in the hardware architecture have been possible because of the acquisition, through the NAIC equipment grant with Texas Instruments, of two TI Explorer Lisp machines. We have designed and begun implementation of a speech research software development environment for the TI Explorer augmented with TI's Odyssey signal processing board. The speech analysis workstation that is developed will be similar to the system we are currently using, a Sun workstation running SpeechTool from SR Systems in Rochester, N.Y. Our new environment will incorporate many of the features of SpeechTool, but it will represent a significant enhancement in flexibility, ease of use, and processing power. Most importantly, the Explorer/Odyssey workstation provides an ideal hardware configuration onto which to map our software architecture.

6.2.1 Software Architecture

The major problem in signal processing applications in general, and speech processing in particular, is the intelligent reduction of vast quantities of data to a manageable amount of useful information that is sufficient to allow interpretation of the signal. Basic to the approach that we have taken to speech understanding is the observation that it is possible for highly trained individuals to interpret a speech spectrogram. Since this is a knowledge based activity that can be explained by the individual doing it, we are attempting to build knowledge based systems that perform the same general task. This philosophy pervades the speech understanding system we are building and is reflected in the system's architecture. The high level architecture is largely unchanged from that presented in last year's report, so we will present only a brief review of the software architecture.

A digitized speech sample is processed by standard signal processing algorithms like FFT and LPC analyses. These data are fed to several parallel feature extraction modules that pull out parameters useful for recognizing intermediate level phonological constituents such as phonemes or syllables. By extracting these features, the utterance is transformed from a continuous signal to a sequence of discrete vectors, where each vector represents the feature values for a short interval of the

utterance. The reduction of data at this level is substantial and results in a representation for the utterance that contains the kinds of features useful for intelligent recognition.

The sequence of feature vectors is then processed by a module that makes broad phonemic categorizations. Adjacent vectors identified as belonging to the same category are grouped into preliminary segments that represent phonological constituents of the utterance. These segments can be thought of as regions of the utterance that are roughly homogeneous, and they represent the results of the first coarse segmentation of the utterance.

The preliminary segments are then examined in greater detail by more specialized modules that refine the categorization and segmentation to label segments with possible constituent identities and likelihoods. This multi-level approach to identifying the phonological constituents of an utterance and assigning confidence factors to those identifications again mirrors the performance of human spectrogram readers and results in an intelligent reduction of data to higher level information.

Once the phonological constituents have been labeled and assigned weights, the utterance has been transformed into a set of alternative sequences of speech constituents. The reliability of the lower level feature extraction and constituent recognition modules determines the size and complexity of that set. This in turn determines the difficulty of the task of the next module, which parses the alternative phonological transcriptions to yield hypothesized word candidates. This is done by using a lexicon of constituent word pairs in conjunction with a knowledge based system that weighs and resolves conflicting hypothetical parses.

At the highest level, a natural language understanding system decides on the most likely syntactic/semantic parse and arrives at a semantic representation for the utterance. This module differs from traditional natural language understanding systems in that the words it receives may not be correct. It must, therefore, be more robust and must consider the confidence factors propagated up from the lower level modules. The final output of the system, then, is the most likely intent of the utterance.

The software architecture can be characterized as pursuing a largely data driven or forward chaining control strategy instead of a more exotic strategy like a blackboard architecture. This decision is based on the assumption that a reasonably accurate phonetic transcription can be assigned to the unknown utterance.

6.2.2 Hardware Architecture

As mentioned above, a great deal of work has been done toward moving the project onto the TI Explorer/Odyssey workstation. The TMS 32020 signal processors on the Odyssey board provide better-than-real-time performance for traditional signal processing like FFT and LPC analyses, and the existence of four of these processors provides a parallel physical architecture onto which to map our parallel software architecture at the feature extraction level.

Preliminary work on the design of the development environment shows, for example, that FFTs, probably the most complex and expensive algorithm we will use, can be computed in less than one-third real time. Our conclusion is that the signal processing and feature extraction algorithms can be computed in real time. This satisfies one of the goals of the system architecture, which is to eliminate the bottleneck at the signal processing and feature extraction levels.

The design of the Explorer/Odyssey speech research development environment is completed and its implementation is currently under way. While the development of this workstation environment is not part of the NAIC speech understanding project per se, its design was influenced by the NAIC project, and its implementation will allow the NAIC project to migrate to the Explorer workstation.

The goals for the environment include the creation of a flexible, easy to use environment that integrates the low level signal processing power available on the Odyssey board with the symbolic processing available on the Explorer to provide speech researchers with a tool processing and studying speech at the appropriate level. The environment's main purpose is to allow users to rapidly build and execute configurations of programs and display the results of those programs graphically. While the first applications built on the workstation will be connected with speech understanding, the intent is to build an environment that will prove useful to speech researchers in a wide variety of application areas.

The environment itself is a largely object oriented system built around four kinds of conceptual objects that make sense to speech researchers: data objects, processes, displays, and application configurations. Data objects are traditional data structures like a raw speech sample, a sequence of FFTs, a feature vector, or a sequence of possible phonemes. They may exist as files, or they may be "virtual" objects created and released by the environment during the course of executing a user's application.

Process objects are programs written in TMS 32020 assembler for the Odyssey or in Lisp for the Explorer. Processes obviously read and create data objects during their execution, and they may be piped together when the output of one process is the input to another. In this case the intermediate data object might not exist as an actual file unless the user wishes to save it for future use. As users implement and debug new programs, they can install them in the environment by describing program attributes and characteristics, which are later used to set up the programs for execution in a particular application.

Displays are graphical windows used to display data objects. Again, the data object being displayed might be a virtual one, as in the case of displaying the output of a program. The environment will know enough about the object being displayed to be able to choose an appropriate manner for displaying it, but the user will be able to alter this default display and do the usual zooming and moving of the various displays while the environment maintains time alignment dependencies among displays of data objects representing different features of the same utterance.

The most interesting object in the environment is the application configuration. This is a directed acyclic graph containing data, process and display objects in a configuration that the user constructs. A graphical programming language allows users to "draw" their applications, which can then be executed by a simple data flow simulator that manages the communication between processes in both the Lisp and Odyssey environments. The system will also manage the parallelism among the Odyssey processors, and it will perform memory management duties for the program and data areas on the Odyssey.

An important benefit of the workstation environment to the speech understanding system is that we will ultimately be able to literally "draw" the software architecture for the speech understanding system as a collection of application configuration objects in the development environment. This means that the development environment will handle the complex mapping from software to hardware architectures in an elegant and flexible way.

6.2.3 Future Directions

Currently, there are two NAIC-funded thesis projects under way on the Explorers, and several more are expected to begin in the spring quarter. The first version of the Explorer/Odyssey workstation environment will be ready by the summer of 1988. This initial version will have the functionality outlined above, but it will lack several finishing touches and enhancements slated for inclusion next year. We also plan to move most of

the speech understanding project off the Sun workstation and onto the Explorers by summer, 1988. By the end of 1988 we plan to begin integrating the various modules developed so far into a prototype of the entire system on the Explorer/Odyssey workstation.

6.3 Feature Extraction

6.3.1 Formant Tracking

Some of the features we would like to extract from the speech signal, such as formants and pitch, are not, strictly speaking, properties of the acoustical signal, but rather are properties of the underlying speech production system. The best way to handle this kind of problem is with machine learning techniques. One disadvantage of such techniques is that they require a considerable amount of training data. To assist in the collection of this training data, we have developed an interactive graphical tool for hand editing formant tracks. We are also working on automating this process with an expert system (see section 6.3.2).

Formants are a rich source of information about the speech signal, but have not been used much in speech recognition systems because of the difficulty in extracting them. Standard peak picking algorithms are too unreliable for this task: although they provide good results most of the time, they are capable of making large mistakes. Such mistakes can lead to serious errors further along in the recognition process. For example, if F1 and F2 merge in the region of a back vowel such as 'uw', the real F3 might be mislabeled as F2, causing the following vowel classifier to identity this region as a vowel closer to an 'iy'.

Our work on formant tracking was motivated in part by some work done by Kopec (1986) on applying Hidden Markov Models to formant tracking. Our approach is similar in spirit to Kopec's, but uses a different statistical technique. We decided to use a classical statistical pattern recognition method from multivariate analysis. The basic idea is to classify an individual formant value into one of a disjoint collection of frequency ranges, e.g., 400-600 Hz, 600-800 Hz, and so on. The classification decisions are made according to a standard maximum likelihood measure. A variety of features can be used in this scheme, including peak frequencies and amplitudes, LPC and FFT spectral values, and other derived measures.

We have completed an initial study of this technique using various features to track F1 and F2. The data used in this study consisted of 38 vowel dense utterances from Carnegie-Mellon, divided approximately equally among 2 adult

male and 2 adult female speakers, for a total of about 95 seconds of speech. The signals were originally sampled at 16 kHz and then downsampled to 12.8 kHz. The basic rate of analysis was 1 frame/msec, yielding a total of about 95,000 frames of data.

The basic system for tracking F2 was as follows. The speech signal was first subjected to 14th-order LPC analysis to obtain the peaks of the LPC spectra. These peaks were then hand edited using a mouse oriented tool to produce correct tracks for the first three formants. An LPC spectrogram of the utterance was available for reference. The resulting tracks were then separated and smoothed, yielding "true" tracks for F1, F2 and F3. A set of features was extracted from each frame of the signal and combined with the true F2 track in computing the necessary training statistics. The statistics for a collection of utterances (all of one speaker, all of one sex, or all four speakers) were then combined to form the final training statistics. For testing, the same features were extracted and used to determine discrete output tracks, the values of which were the midpoints of the frequency ranges (e.g., 500 Hz, 700 Hz, etc.). These tracks were then smoothed using a combination of median smoothing and a moving average to produce the final output track. Figure 6.1 illustrates this process for the utterance "The angry crowd pushed open the doors" spoken by a male speaker.

Our first idea was to use peak information for the features to be extracted from each frame. This works reasonably well for F1. The peak related features that were found to perform best for F1 were the frequencies and amplitudes of the first two LPC peaks. When a quantization size of 200 Hz was used, the following results were obtained. 74.5% of the frames were correctly classified, the RMS error was 46.9 Hz, the mean error was 32.5 Hz, 1.7% had errors greater than 100 Hz, 0.1% had errors greater than 200 Hz, 0.9% were missed formants, and 17.2% were false alarms. The false alarm rate for most tests turned out to be rather high due to the conservative nature in which the hand editing was performed, so we also determined the percentage of false alarms which were not adjacent to a correct track. For the F1 results above there were only 1.4% such false alarms.

The majority of our time on this study thus far has been spent on F2, which is a more interesting case. For comparison, we tested a simple peak picking algorithm (Markel, 1976) on our database. This algorithm performs well most of the time, but it does make some large errors. In fact, 76.9% of the frames tested were within 50 Hz of the correct values, but 1.3% had errors greater than 1000 Hz. In addition, the RMS error was 235.8 Hz and the mean error was 93.7 Hz. A more detailed

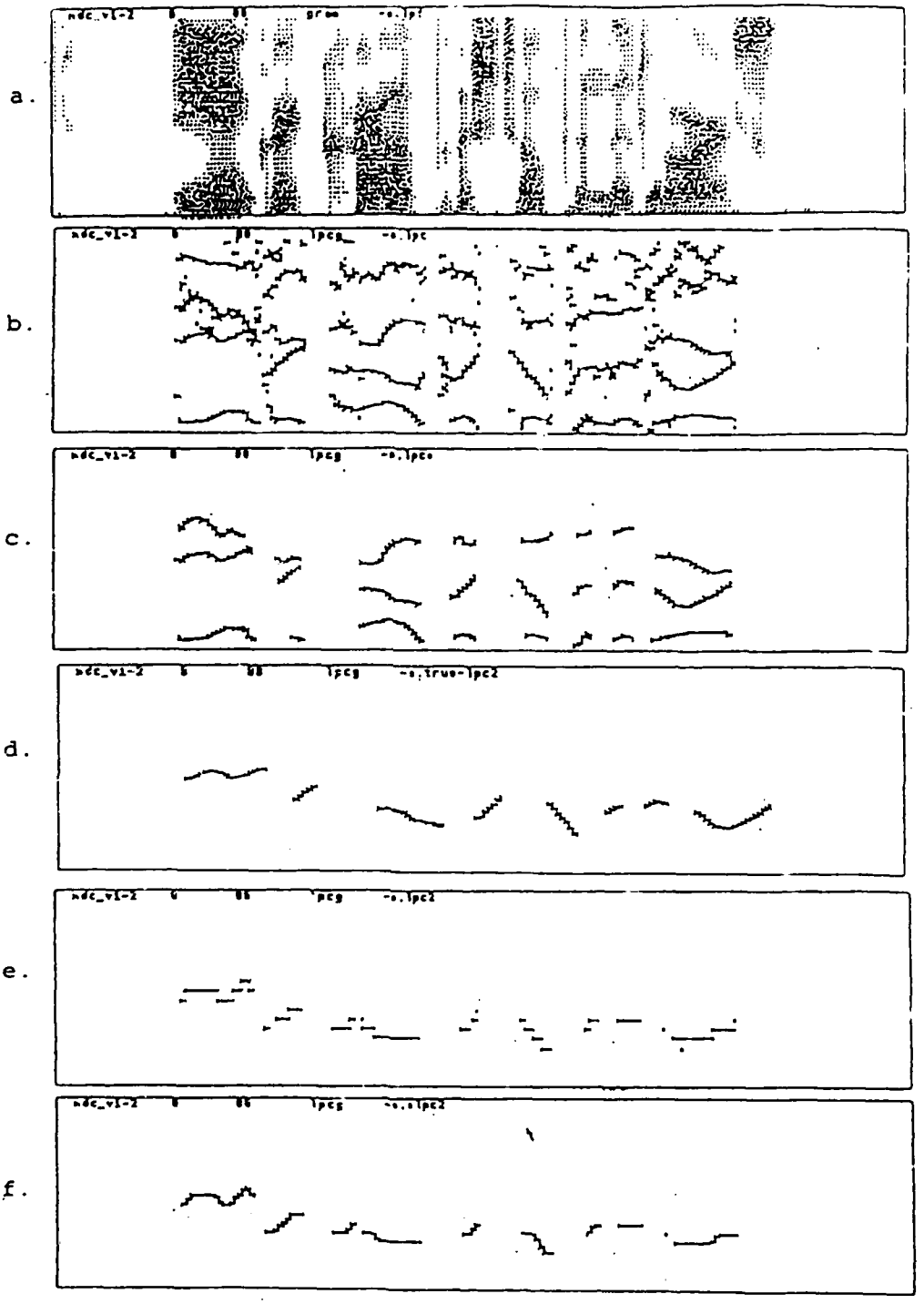


Figure 6.1 (a) LPC spectrogram of "The angry crowed pushed open the doors", (b) LPC peaks; (c) hand-edited LPC peaks, (d) smoothed F2 trajectory from hand-edited LPC peaks, (e) discrete output from formant tracker using six averaged spectral peaks and trained on the entire database, and (f) final smoothed output.

examination of the results from these tests reveals that the particular algorithm used here performs much better for males than females, but that large errors are made for both sexes.

Peaks turned out to be poor features for our system when tracking F2. Instead, we focused on using averaged spectral values. We found that the most useful values were between 400 and 3000 Hz, which is slightly beyond the range of the F2 values in our database. LPC spectra were averaged first over time to provide continuity and then over frequency to produce a fairly small number of features. The results indicate that with six such features, reasonable performance is obtained in the speaker specific and sex specific cases, but not overall. When testing and training was performed on a particular speaker, the RMS error was 68.8 Hz, 0.8% of the errors were greater than 200 Hz, and none of the errors were greater than 500 Hz. When the testing and training was performed on both males, the RMS error was 82.7 Hz, and for females it rose to 85.6 Hz. For the entire database, the RMS error jumped to 143.1 Hz. This was due primarily to a few large errors. Better results can be obtained by adding more features, although more computation time is required, of course. Using thirteen averaged spectral values, the RMS error was 61.7 Hz for a single male speaker, 68.7 Hz for both males, 75.6 Hz for both females, and 93.1 Hz for the entire database. With these thirteen features the number of errors greater than 200 Hz was at most 2%, and there were no errors greater than 500 Hz.

6.3.2 An Expert System for Formant Labeling

A number of our research projects require accurate, hand edited formant frequency data. For example, in order to test the formant tracker, it is mandatory to compare its output with correct data. Unedited files resulting from LPC analysis contain occasional spurious peaks, merged formants, etc. Therefore, it is necessary to hand label each file. That is, an expert views the signal as a spectrogram, a plot of the LPC peaks, and a label file containing the identification of each phoneme as well as the starting and ending frame for that phoneme. With the knowledge of the frequency ranges for each of the phonemes, the sex of the speaker, and heuristics about the influence of certain phonemes on other phonemes, the expert is able to make intelligent decisions as to which peaks are actually formants. This is very time consuming and costly. Therefore, the purpose of this project is to develop an expert system to replace the expert or, at the very least, reduce his workload. This study will include the labeling of monophthongs, diphthongs, glides, and liquids. It will look at the problem associated with nasals, but at this time they are not included in the study.

The input to the system consists of a minimum of four frequency-amplitude pairs for each frame. These are obtained from the raw signal which is transformed into peaks with corresponding amplitudes using linear prediction. Linear prediction is primarily a time domain coding method, but can be used to give frequency domain parameters like formant frequency, bandwidth, and amplitude. It uses the present speech sample as a prediction of the next one and stores the prediction error, that is the sample-to-sample difference.

The first part of the linear prediction process used in this study involves applying the Hamming window, a special purpose weighting function used for reducing distortion in the frequency domain. Then the autocorrelation method is applied to calculate the linear prediction and gain. The output produced is six frequency-amplitude pairs for each frame. These six frequency values, peaks, are called the first six formants, making the assumption that a pole strong enough to show up as a peak is a formant.

The approach taken is a combination of knowledge engineering techniques to simulate the behavior of the human correcting the files, and an algorithmic approach to accomplish what the human does with visual cues that cannot be coded into the computer. From the phonetic transcription, both the expert and the expert system identify the next target phoneme. In addition to this, the neighboring phonemes and the sex of the speaker are also known. Using a rule based approach and the above knowledge, the system is able to determine the best starting frame and a starting estimate for that phoneme.

The expert now uses the visual cues of the LPC peak graph to connect the peaks of the LPC spectrogram. Continuity is one of the strongest constraints one can rely on in tracking formants. Because the visual cues are not available to the computer, an algorithm by McCandless (1974), slightly modified to meet the needs of the system, is employed. Starting at the "pick" frame and working forward and then backward, four formants for each frame are dealt with together. The output only requires three formants; however, four points are often needed to calculate these three values.

The algorithm begins with four vacant formant slots, four estimates for the frequencies of the formants, and possibly four peaks (it is conceivable that some peak slots will be empty). Each formant slot is filled with the peak value closest in value to its corresponding estimate. The peaks that are used are so marked. If more than one slot has the same peak value, any duplicates are removed by determining the best slot for peak. If a peak is still unassigned, it is assigned to the

corresponding or neighboring slot if that slot is vacant. For each slot that is assigned a value, the corresponding estimate is updated. If a slot has no value, the corresponding estimate is left as is. The answers (values of the slots) are recorded, and the next frame is processed.

After all frames for a particular phoneme have been processed, the system checks the data using information about the phoneme and neighboring phonemes to determine if it is confident in the results, or whether the segment should be hand labeled. The file is then smoothed, and the next target phoneme is processed.

To date, the rule based approach to determine the best starting frame and first estimate for each of the four formants works: the system's choice for a starting frame and estimates is either the same or close enough to the expert's that no difference is noticed in the manner the rest of the system behaves. The major problem encountered so far is the deletion of an entire set of data points for a particular formant because the values are out of range for that phoneme. This occurs when the phoneme is next to a liquid, glide, or nasal. The current plan is to address this problem by incorporating information about the influence of these phonemes on the target phonemes as well as information about the amplitude of the data points.

The final step in development will be to add diphthongs, liquids and glides to the database. Diphthongs have to be treated separately because they are actually two vowels connected by a transition, resulting in very strong contextual influences. Liquids and glides have to be treated somewhat differently because there is a greater degree of vocal tract constriction than for vowel elements which will influence the behavior of the formants. After a final smoothing, the system will determine if the results are reasonable based on information in the database about that phoneme in that context.

6.4. Phonetic Classification

6.4.1 Vowel Recognition in Static Environments

One of the most basic problems in speaker independent phonetic analysis is to develop a phonetic space that is minimally sensitive to differences from one speaker to the next; that is, to find a phonetic space in which acoustic representations of a particular phonetic category look similar across speakers. The purpose of the study described in this section was to develop such a space for the representation of vowels.

A very large number of schemes have been proposed in the

literature for classifying vowels produced by different speakers based on some combination of fundamental frequency and formant frequencies. The present study was designed to compare these normalization schemes against one another using the same classification technique and the same data base.

The data base that we have been working with is a set of measurements made at Bell Laboratories by Peterson and Barney (1952). The data set includes hand measurements of fundamental frequency and the three lowest formant frequencies of 10 vowels produced by 33 adult males, 28 adult females, and 15 children. The vowels were produced in the [h-V-d] environment.

The classification technique that was used is a maximum likelihood distance measure, which is very similar to the Mahalanobis (1936) measure that was used in a recent study by Syrdal and Gopal (1986). The only difference between the technique used by Syrdal and Gopal and the one used in the present study is that the Mahalanobis measure uses a covariance matrix that is pooled across all of the training categories. The maximum likelihood distance measure employed in the present study uses a separate covariance matrix for each training category. This has turned out to be an important distinction because we have found a few cases in which maximum likelihood technique works significantly better than the method that is based on a pooled covariance matrix.

The results that are shown in Table 6.1 all make use of 'internal' or 'speaker independent' parameter sets -- these are parameter sets that make use only of information that is available in the unknown token, without any information that describes the individual speaker. The parameter set is shown on the left; the numbers in the table represent overall classification accuracy across all talkers and all vowels. Results are shown for linear frequencies, log frequencies, bark scale transforms and mel scale transforms.

Classification accuracy with F1 and F2 alone is about 75-76%, no matter whether the frequencies are represented in linear dimension, or whether some kind of nonlinear transform is used. Adding F3 helps, but adding f0 helps even more. Again, the transform to a log, bark or mel scale does not seem to make much difference. It can also be noticed that once f0 has been added to the parameter set, the addition of F3 produces only a very small improvement in classification accuracy.

The last set of entries in the table are for calculations based on spectral distances rather than absolute frequencies. Miller and his colleagues (Miller, Engebretson, and Vemula, 1980; Miller, 1984; Fourakis and Miller, 1987) have been working with a vowel classification scheme based on three

Table 6.1. Classification results for internal parameter sets.

Parameter Set	linear	log	bark	mel
F1, F2	74.9	76.8	75.9	75.2
F1, F2, F3	84.2	84.2	84.4	84.3
F0, F1, F2	86.1	85.6	85.1	86.3
F0, F1, F2, F3	87.7	87.3	87.5	87.3
F1-F0, F2-F1, F3-F2	86.3	86.9	87.3	86.1

parameters: (1) logF3-logF2, (2) logF2-logF1, and (3) logF1-logf0. The classification system that was proposed by Syrdal and Gopal (1986) is very similar, except that spectral distances are represented on a bark scale instead of a log scale. As table 6.1 shows, our results do not show any advantage for an approach that is based on spectral distances rather than absolute frequencies. It can also be seen that there was no advantage for any of the nonlinear scales over a linear frequency scale.

The data that are presented in Table 6.2 represent results for parameter sets in which internal information has been combined with some generic information that describes the individual talker -- information like average f0, average F1, etc. Again, the parameter set is on the left and overall classification accuracy is on the right -- 'mf0' is the mean f0 for the speaker, 'mF1' is mean first formant frequency, and so on.

The first set combines F1 and F2 with mf0, mF1, mF2 and mF3. The main point is that mF1 and mF2 work better as normalizing parameters than either mf0 or mF3. The next set of tests are identical, except that F3 has been added to the parameter set. The pattern is the same, but the overall performance is slightly higher when F3 is added to the set.

The last set of tests represent results for several different combinations of normalizing parameters. The main point here is that a combination of two or more normalizing parameters performs better than any single parameter by itself. It is interesting that you can do quite well without either F0 or F3 if the right set of normalizing information is included. In general, though, the more parameters you use, the higher the performance, although the differences among the various parameter sets here are not especially large.

From our perspective, in terms of applying these results to automatic speech recognition, the most attractive parameter set in this table is the first entry in the bottom set -- F1, F2, mF1, mF2. The performance is reasonably good, and everything can be accomplished by measuring just two parameters -- F1 and F2. Higher classification accuracy can be achieved with more parameters, but the relatively modest improvement in accuracy would almost certainly be offset by the greater chance of making formant tracking error when you try to track three instead of two formants.

Below are the main conclusion from this study.

1. We found no evidence that nonlinear transforms produced better classification accuracy that linear frequency scales.

Table 6.2. Classification results for parameter sets in which internal information is combined with generic speaker information.

Parameter Set	Classification Accuracy
F1, F2, mf0	85.8
F1, F2, mF1	88.0
F1, F2, mF2	88.3
F1, F2, mF3	85.0
F1, F2, F3, mf0	87.8
F1, F2, F3, mF1	89.5
F1, F2, F3, mF2	89.5
F1, F2, F3, mF3	88.0
F1, F2, mF1, mF2	90.8
F1, F2, mf0, mF1, mF2	91.0
F1, F2, mf0, mF1, mF2, mF3	91.1
F1, F2, F3, mF1, mF2, mF3	91.5
F1, F2, F3, mf0, mF1, mF2, mF3	91.8

Clearly these nonlinear transforms -- especially the bark and mel-scale transforms -- make much better sense than linear frequency in terms of what we know about the physics and pyschophysics of the auditory system. But that fact does not, by itself, mean that these transforms will necessarily solve any problems related to speaker variability.

2. We found no evidence that parameter sets based on spectral distances performed better than parameter sets based on absolute frequencies. If anything, absolute frequencies seemed to do a little better.

3. A significant improvement in performance can be achieved by combining internal information with a small amount of generic information that describes the individual speaker. Average F1 and average F2 seem to be the most effective normalizing parameters. A very simple parameter set consisting of F1, F2, mF1, mF2 correctly classifies about 91% of the vowels in the Peterson and Barney data set.

The next step on this project will be to apply some of these normalization techniques to vowels in continuous speech (described in Section 6.4.2), and to test performance using automatically extracted formant frequencies.

6.4.2 Vowel Recognition in Dynamic Environments

The research that is described in this section is an extension of the work in section 6.4.1 above using the maximum likelihood multivariate distance measure (MVD). The MVD work described earlier was limited to a single spectral slice, and while it produced excellent results, there was an important limitation. The data used for the MVD were hand measured and thus eliminated the question of where along the spectrum the features best describe each phoneme. In order to develop a practical vowel recognition system it is necessary to analyze patterns whose feature values change over time.

The basic idea of this research will be to sample the features several times throughout the course of a speech sound rather than relying on a single spectral slice. This approach allows the system to account for dynamic spectral information, and eliminates the need for decisions to be about a single spectral cross-section to represent the speech sound. The MVD technique can then be used to categorize the phrase by determining the distance of the unknown segment to the library of reference segments.

A new problem which arises is the time alignment of the segment to the library of reference segments. Fortunately,

there has been an important breakthrough in the past few years by the development of the Dynamic Time Warping technique. This process attempts to time align the segments by compressing portions of each segment to reduce the overall sum of distances between the two segments.

A category which the vowel recognizer will attempt to partition a segment into is called a class. A class can be as small as a phoneme or as large as a syllable. While phonemes might be the smallest number of classes, they may not produce acceptable results because of the context effects that result from coarticulation. Syllable classes would provide good results, since they take into account the context of the central vowel with the surrounding consonants. Unfortunately, due to the large number of syllables in the English language (approximately 10,000) the data base would be unmanageable. It is felt that some abstract class between the phoneme and the syllable would produce the best results.

This project will attempt to use syllable classes, providing the best partitioning of classes, in an attempt to concentrate on the development of the vowel recognition system itself. Work on reducing the set of classes can be deferred due to the relative ease with which the system can incorporate a new set of classes. One need only retrain the system with the new classes, since the training and recognition portions do not require any other information other that how the phrases are segmented using the set of classes.

The data base that will be used in the vowel recognition was obtained from Carnegie-Mellon University. The speech corpus is a vowel dense data base containing 35 male and 35 female speakers. The data has been hand segmented and a phonetic transcription of all utterances was included.

The phonetic features that will be used in this project will be the first and second formant frequencies (F1 and F2) and for each speaker the mean of their first and second formant frequencies (mF1 and mF2). As was discussed in section 6.4.1, this simple feature set produces accurate identification for the static case (Hillenbrand and Gayvert, 1987).

6.4.3 Human Benchmarks

Peterson and Barney's (1952) classic study demonstrated that vowels in CVC context produced by men, women and children are identified with a very high degree of accuracy by human listeners. Several attempts have been made to classify the Peterson and Barney vowels using various combinations of F0 and F1-F3. The work described in Section 6.4.1 was designed to compare the performance of a large number of vowel

classification schemes using the Peterson and Barney measurements and a maximum likelihood distance measure. The most successful internal parameter set, which consisted of F0 and F1-F3, correctly classified 87.7% of the Peterson and Barney vowels.

It is very important to note that the performance of even the best classification algorithms has been significantly below that of human listeners: the 94.4% accuracy shown by Peterson and Barney's listeners has not been approached by any automatic classification algorithm that does not make use of speaker dependent normalizing information. However, Peterson and Barney's listeners had access to dynamic (duration and spectral change) as well as static information (F0 and "target" formant frequencies). The purpose of the present study was to determine the identifiability of the Peterson and Barney vowels based exclusively on static information.

A formant synthesizer (Klatt, 1980) was used to generate 300 msec steady-state versions of all 1,520 signals (33 men, 28 women, 15 children X 10 vowels X 2 repetitions) in the Peterson and Barney data base using measured values of F0, F1, F2 and F3. Twelve subjects with varying amounts of phonetics training participated as listeners. As in the original study, stimuli were presented in blocks of trials, with 10 talkers per block (4 men, 4 women and 2 children).

Identification results for steady-state versions of the Peterson and Barney vowels are shown in Table 6.3, along with data from the original study. The 24.8% overall error rate for the steady-state, resynthesized stimuli is more than four times greater than that for original stimuli, and twice the error rate for an automatic classification algorithm based on F0 and F1-F3 (Hillenbrand and Gayvert, 1987). The error rate was the lowest for male talkers and highest for child talkers, a pattern which was shown by all 12 listeners. Error patterns for specific vowel categories showed some similarities to the original data (e.g., exceptionally good performance on "IY"), but there were also some important differences (e.g., the large difference in performance between "IY" and "ER").

One very clear implication of these results is that there is a good deal of information in the static spectral cross-sections that formed the basis for the steady-state stimuli. The static spectral patterns provided enough information for subjects to identify 75% of the stimuli, and to select an adjacent vowel category on most of the trials in which errors occurred. However, the significant increase in error rate compared to the original signals indicates quite clearly that there is a great deal of information missing from the steady-state signals. This finding suggests that human

Table 6.3. Percent correct identification of steady-state resynthesized versions of the Peterson and Barney vowels. For comparison, identifcation results are shown for the original stimuli.

Vowel	Steady-State Resynthesis	Original Stimuli
IY ("heed")	95.1	99.9
IH ("hid")	66.8	92.9
EH ("head")	71.1	87.7
AE ("had")	72.9	96.5
AH ("hod")	70.5	87.0
AW ("hawed")	74.0	92.8
OO ("hood")	67.9	96.5
UW ("who'd")	82.6	99.2
UH ("hud")	71.7	92.2
ER ("heard")	79.8	99.7
Total:	75.2	94.4
Males:	77.1	*
Females:	75.2	*
Children:	71.2	*

*Peterson and Barney did not report results separately for male, female and child talkers.

listeners make use of dynamic information such as duration and spectral change in identifying vowels. How this dynamic information is mapped onto phonetic categories is not well understood. Follow-up studies will explore this question as well as a broad range of other issues related to these results, including an examination of the roles played by fundamental frequency, duration, and spectral change.

6.4.4 Future Directions

Work to date has focused on the development of an acoustically based feature space that allows classification of resonant sounds (vowels and diphtongs) across speakers. A major goal of future work will be to extend this learning based approach to other classes of speech sounds, and to combine the phonetic classification techniques to the feature extraction and segmentation modules described in other sections of this report.

6.5 Broad Phonetic Classification

6.5.1 Overview

Feature extraction is the first step in the analysis process encountered by the speech signal. During this step characteristics such as, zero crossing rate, total energy, formant frequencies, etc. are extracted and a feature vector is produced. This vector is the collection of all these features for a particular sample period of the signal. The next step, coarse classification, involves assigning sequences of feature vectors to one of five broad phonetic categories (fricative, vowel, stop, silence, and other).

After broad classification, the general class of the phoneme has been hypothesized and its end points are roughly marked. Each section of speech is then presented to the appropriate specific phoneme identifier. Each of the specific phoneme identifiers are responsible for an in-depth analysis of its particular phonetic class.

This project will concentrate on coarse classification. There are two major objectives for the work outlined in this section: (1) to examine the merits of K-means clustering, a tree structured architecture, Mahalanobis distance, and linear distance measures as they apply to coarse classification, and (2) to produce a working version of a coarse classifier. The remainder of this section will discuss the details of how these objectives will be reached.

6.5.2 Training

Training involves systematically examining many samples of speech data in order to allow the system to learn the salient characteristics of the category it is trying to classify. The training process used in this work will be examined in three steps.

1) Feature extraction and data preparation
2) Cluster analysis
3) The output of the training phase

In the feature extraction and data preparation phase, raw samples of speech data will be examined to produce a collection of label/vector pairs (LVP). A LVP is a two-part entity consisting of a label, which is taken from hand labeling information in the data base, and a feature vector. The feature vector is an ordered set of integers, each integer corresponding to a particular characteristic of the speech sample. The features being used are zero crossing rate (ZER), total energy (TOT), relative energy (REL), peak energy (PEA), spectral derivative (SPE), and history (HIS).

The objective of this first step of the training phase is to generate a collection of LVPs that will form the basis for the cluster analysis. Four kinds of data files are involved in the creation of LPVs: CMU speech data base, the options file, the executable code, and the resulting output files which is the collection of LVPs.

CMU Data Base. The speech data base used for this thesis was supplied by Carnegie-Mellon University. It includes 371 utterances, of which 143 are rich in fricatives, 129 in stops, and 99 in vowels. The data base was spoken by 35 different speakers and has been hand labeled with phonetic transcriptions. The speech data is 12 bit linear PCM sampled at 12.8 kHz. The CMU data base will supply both the training and testing data for this work.

Output File. The options file contains information the preprocessor uses in order to extract and prepare the data properly. A brief description of this information is presented below.

SOURCE = file name
: File name of a speech utterance to be used in this training session. One or more of these lines can be specified.

FEATURE = feature list

Specifies what features are to be extracted from the speech utterance. One or more of these lines can be specified, each with one feature listed from the following; ZERO_CROSSING, TOTAL_ENERGY, PEAK_ENERGY, RELATIVE_ENERGY, SPECTRAL_DERIVATIVE, HISTORY.

PHONEME = phonemes [= num]
Specifies which phonemes are to be used. Each input file has a phonetic transcription file and the preprocessor will only extract features from phoneme types specified here. The phonemes are CMU type names separated by commas. If a "*" is specified then all phonemes are used. The second equal sign followed by a number is optional and indicates how many samples of the specified phonemes the preprocessor is to use. If omitted the preprocessor will use all samples of the phoneme it finds in the source files. One or more of these lines may be specified.

FRAME = msec
Specifies the length of the frame in milliseconds. If not specified this will be 10 msec.

Preprocessor. The preprocessor section includes all the executable code needed to extract the LVPs. This code will consist of a collection of programs that perform specific operations on the speech data and a final program that will organize the results and produce the appropriate output files.

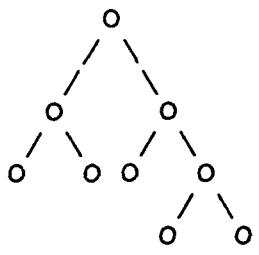
Output. Seven files will be produced: one for each of the six features being extracted, and one containing the phonetic label information.

6.5.3 Cluster Analysis

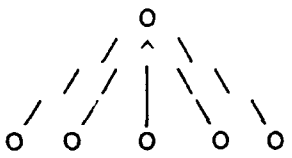
Once the feature information has been extracted and the LVPs collected, the clustering can begin. This involves assigning each LVP produced in step 1 to one of the five coarse classes. The collection of all the LVPs assigned to any one class is called a cluster. The goal of this phase to split the collection of LVPs into five clusters where each cluster represents one of the five classes being trained.

The determination of which class an LVP belongs to can be done in many ways. No matter what technique is used there must be an underlying structure to the decision making process. This structure can be as simple as looking at all the features together and making one single decision or it can be broken down into a series of smaller decisions looking at a subset of the features during each decision. It is this latter structure that will be used in this work and this structure can best be

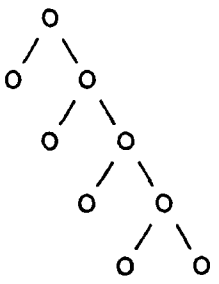
represented by a tree, where each node represents a cluster and the branches coming from a node represents the decision that must be made. The three trees, shown below, represent the extremes of the tree structures for five final classes and will be compared in this work.



Binary Tree



Single level tree



Skewed binary tree

With the tree structured decision process, the goals of the clustering phase are to determine the clusters of LVPs at each node in the tree, and to determine which features should be used at each of the non-terminal nodes. Two techniques will be used to address these problems.

The simplest and most obvious clustering technique is to place the LVPs into one of the five terminal nodes based on the phonetic label. This will be done by taking each non-terminal node and using a Mahalanobis measure to split the cluster between its children using every combination of features. For example, if there are three features (f1, f2, and f3), then this loop would execute seven times, once each with the following feature sets; [f1], [f2], [f3], [f1,f2], [f1,f3], [f2,f3], [f1,f2,f3]. The feature set showing the fewest errors will be used for the later classification of an unknown frame.

The second approach to clustering uses the feature vector itself to separate the data into clusters. The K-means algorithm, described earlier, will be applied to perform the clustering. All the LVPs are initially in the root node cluster. The K-means clustering algorithm will then separate the LVPs between the children of the root node. The process will then repeat itself on each of the children until all the feature vectors are assigned to one of the five terminal nodes. At each of the non-terminal nodes the K-means clustering algorithm will be run once for each combination of features. The performance index will be the percent of feature vectors correctly classified as specified by the phonetic label. A difference between this and the previous clustering method is that although this technique is using the phonetic label to determine the performance of the features, it is not using the label to determine the final destination of the LVP. The final destination node is based solely on the feature vector. The

pseudo code below outlines this process.

```
Procedure MainKmeans;
   Read in tree structure;
   Associate all feature vectors to the root node;
   Call TrainKmeans( RootNode );
End;
Procedure TrainKmeans( Node );
   If Node is a terminal then
      done;
   else
      For each feature set do
         Perform K-means clustering; /* K=number of children
                                        from this node */
         Get a performance index for the feature set;
      end;
      Choose the best performance index;
      Report the feature set used and clusters;
      Assign each feature vector to one of the children;
      For each of the children do
         TrainKmeans( Childnode );
      end;
   end;
End;
```

Summarizing the results of the training phase, we plan to train the system on three different tree structures, each using two different techniques. This gives the following six combinations of training results that will be compared with each other.

1) Mahalanobis - Binary tree
2) Mahalanobis - Single level tree
3) Mahalanobis - Skewed binary tree
4) K-means - Binary tree
5) K-means - Single level tree
6) K-means - Skewed binary tree

For each combination of techniques, the training session will produce the characteristics of the clusters at each node in the tree, with the exception of the root node. For each of the first three training sessions the characteristics will include a mean feature vector and an inverse covariance matrix. For the last three training sessions the characteristics will be only the mean feature vector.

6.5.4 The classification of an unknown data sample

Once the training is done the classification stage is rather simple. To classify an unknown sample of data the system will perform the following steps:

1) Start at the root node.

2) Compute the feature vector of the unknown speech frame.

3) Calculate the distance from this point to the cluster center associated with each of the children.

4) Then move down in the tree to the child whose cluster center is closest to the unknown frame.

5) If we are now at a terminal node then this determines the coarse class of the phoneme. If we are not at a terminal node then return back to step 3.

6.5.5 System evaluation and results

Evaluation will be accomplished by giving all six systems the same set of utterances from the CMU data base and letting them produce the coarse phonetic transcription. The results will then be compared to the hand labeled phonetic transcriptions to determine tha accuracy of each classifier. These tests will be repeated with utterances both from the training set and from outside the training set. Additionally an analysis of the incorrectly placed frames will be done to determine what types of errors were made and recommend ways to either avoid these errors or detect and correct them after they have occurred.

6.6 Phonetic String Parser

6.6.1 Background

As the result of previous processing on several levels, an unknown utterance is transformed into a string of undifferentiated phonemes (i.e., no word boundary markers). The phonetic string parser scans this sequence of phonemes, hypothesizing all words in the utterance that are consistent .ith the lexicon. Generally, this can be accomplished by comparing reference patterns in a phonemic lexicon with the unknown sequence. All words hypothesized are then passed to a syntactic and semantic parser for further analysis.

However, there are three areas of complexity which prevent the lexical access procedure from being a simple lexicon lookup. Front-end errors, and the effects of phonological recoding, are two areas which alter the symbolic representation of an utterance. Ambiguity that results in multiple parsings from a single phonetic representation is the third area.

It has been observed that the front end of a speech understanding system will at times exhibit an inability to

distinguish between similar sounding phonemes. As a result of this confusability, the string of phonemes representing an unknown utterance may contain insertion, deletion, or substitution errors. Cohen et al. (1975), Klatt (1975), and Oskika et al. (1975), show that in continuous speech there are variations in pronunciation (especially across word boundaries) that are not random and can be described by a set of phonological rules. Finding lexical search methods that compensate for phonological recoding has not proven to be a simple matter. Phonological rules that apply within a word boundary can be handled by creating an alternative base form (i.e. an idealized pronunciation) entry in the lexicon. However, a more difficult problem arises when working with variations across word boundaries. For example, the utterance "did you" may actually be realized as "di-ja", illustrating a deletion and insertion error. Adding another baseform to the lexicon for the word "you" (starting with the palatal "j"), could provide for an erroneous recognition of "you", when the utterance may in fact be "judge". Even with this potential problem, some previous recognition systems such as HWIM (Wolf et al., 1977), SPEECHLIS (Woods, 1975), and Rudnicky et al.'s (1987) lexical access system at Carnegie-Mellon have built lexicons where each word may have alternative representations. These are generated by applying phonological rules to dictionary base form representations.

It is also necessary to consider the potential problem of a single phonetic string with multiple interpretations as a sentence. Ambiguous parsings can map a single phoneme sequence into different strings of words. Matching against entries in the lexicon will not contribute to solving this problem. There is a need for syntax and semantic knowledge to differentiate meanings. This is outside the scope of the phonetic string parser.

6.6.2 Two Approaches To String Comparison

The two basic approaches to string comparison in isolated speech recognition are Hidden Markov Models (Levinson et al., 1983) and Dynamic Time Warping (Itakura, 1975). Both methods operate on the general principle of dynamic programming in searching for optimal paths. Although both approaches have been extended into the domain of continuous word recognition (Baker, 1975; Wolf et al., 1977; Lowerre, 1980; Myers, 1981; Ney, 1984; Levinson, 1987) it is not certain if they are extensible to large vocabulary, speaker independent, continuous speech understanding systems.

The use of Hidden Markov Models (HMM) is a probabilistic technique to model a stochastic process (Markov sources) that is not directly observable, but can be examined through the output

of a sequence of symbols (Rabiner, 1986). Phonemes and words can be thought of as the observed output dependent on the probabilistic changes (transitions) in acoustic signals and phonemes respectively. Through the use of training utterances and empirical observation of a system's front-end performance, one can model the process of phoneme and word generation, taking into account front-end errors, speaker variation, coarticulatory and phonological recoding effects. Words within the unknown utterance would be hypothesized as those whose models had the greatest probability of generating the observed phonemes. The Dragon system by Baker (1975), incorporated the concept of chaining Markov processes in a hierarchical fashion, not only on a word basis, but at the phrase and sentence level as well. The result was a finite state network of Markov sources in which the recognition procedure looked for an optimal path of transitions that would most likely account for the observed utterance.

Dynamic Time Warping (DTW) is a method of sequence comparison, derived from a time sampling of some quantity that is subject to variations. DTW has been successfully used in isolated word recognition by Itakura (1975) and Waibel (1981), in addition to connected word recognition by Myers et al. (1981), Ney (1984), and Watari (1986).

An unknown utterance and a reference utterance should be thought of as two sequences of feature vectors or tokens (phonemes). Each sequence defines the axis of a matrix mapping the speech utterances against one another. At each coordinate is a measure of distance or dissimilarity between the tokens. The goal is to find a path between endpoints of the two sequences whose distance D is minimal. This cumulative distance can then be used as a decision criterion for recognition (Ney, 1984).

Applying the principle of optimization from dynamic programming concepts (Ney, 1984), a recurrence relation minimizing the number of points considered at any one time follows:

$$D(ai,bj) = \min \begin{cases} d(ai-1,bj) + w(ai,0) & \text{deletion of ai} \\ d(ai-1,bj-1) + w(ai,bj) & \text{subst. of ai by bj} \\ d(ai,bj-1) + w(0,bj) & \text{insertion of bj} \end{cases}$$

Weighting coefficients are added to penalize for deletions, substitutions and insertions. However, searching all possible

paths is computationally expensive. Constraints that restrict this search include controlling the degree of slope allowed in the warp, and setting some maximum permissible path distance. These can be used to prune paths that would otherwise grow excessively large.

The Hidden Markov Model is a recognition method that requires the collection of empirical statistics that describe the response of the recognition system's front-end. The determination of states, transitions, and associated probabilities is a complex optimization problem. More significant is the fact that our front-end is not complete, precluding any statistical evaluation. Dynamic Time Warping, on the other hand, need only receive the strings for comparison and the provision of some distance metric. The drawback to DTW is that it requires a large number of distance calculations. A study by Levinson (1983) estimated that HMM, which uses a simpler likelihood evaluation function, requires an order of magnitude less computation time than DTW. Also noted was that both systems achieved comparable error rates.

6.6.3 String Parsing Using Dynamic Programming

Central to the method of DTW is the necessity of some distance measure. Once this measure has been determined, the process of sequence comparison can proceed. Studies by Miller and Nicely (1955) demonstrated that humans typically confuse particular consonants in a consistent fashion. Predictable confusability patterns are also exhibited by the acoustic-phonetic modules of speech recognition systems. Ideally, we would make use of the front-end's response characteristics in classifying all phonemes as a distance measure. However, the vowel classification study by Hillenbrand and Gayvert (1987) is the only portion of the front-end for which data exits. The remainder of the data will be extracted from studies of human confusability by Shepard (1980) and previous phoneme distance measures in a study by Picone et al. (1986).

The lexical knowledge source for this study is the vocabulary taken from a United States Air Force Cockpit Natural Language study by Lizza et al. (1987). The study provides a vocabulary of 656 words, their frequency of occurrence, contextual use, and the number of times a word is preceded or succeeded by other words. This information may be valuable in determining the types of contextual effects to expect. Using a text-to-speech synthesis system (DECtalk - manufactured by Digital Equipment Corporation) in combination with hand coding, phonetic transcriptions have been produced for all words in the vocabulary.

Similar to other level building algorithms (Sakoe and Chiba, 1979; Myers et al., 1981; Ney, 1984) the DTW procedure will move from left to right, finding the collection of reference patterns whose global (phrase) DTW distance is at a minimum over the concatenation of local (word) DTW minimums. A phoneme reference pattern is selected from the lexicon (its selection method will be discussed below) and time warped with an initial portion of the unknown utterance, producing a time normalized distance. This procedure is applied repeatedly to the same section of the unknown, until all acceptable word hypotheses are determined. Hypotheses that exceed a preset minimum distance threshold during DTW calculations, or deviate from other constraints could be pruned early. This reduces the expenditure of computational resources by eliminating otherwise wasteful calculations. However, this threshold cannot be determined without examining the general performance of the DTW process. One might accept all final hypotheses or, based on system performance, rank order them and select a subset. Once a set of hypotheses is generated, the position in the unknown utterance corresponding to just after the end of each hypothesized reference, becomes a starting point for continued DTW analysis. When the process terminates, an analysis is done to produce a set of overall hypotheses whose total accumulated distance is minimum.

Note that, given an utterance of fixed length, and given an equivalent distance between all reference and unknown patterns, a small number of large words will have less total accumulated distance (globally) than a larger number of small words, indicating a possible heuristic that favors use of large reference patterns for DTW prior to smaller patterns. Smith et al. (1980), suggested that large words should be hypothesized prior to smaller words since larger words usually contain more syntactic and semantic value.

Exhaustive search of all lexical entries is not practical with lexicons numbering in the tens of thousands. However, the front-end of our system is not complete, and a benchmark of system performance needs to be established. Therefore, a brute force search technique will be initially implemented, allowing an examination of the algorithms response relative to insertion, deletion and substitution errors. Observations will also be made of how adjustment of the distance threshold relates to different levels of error. As the general performance of the DTW procedure is established, then methods can be introduced to more selectively guide the selection of reference words.

Possible ideas that would constrain the number of reference patterns to be used for DTW include (1) Reference patterns of the lexicon organized as a hash table, indexed by the first phoneme that could be encountered in all reference

representations. During DTW, the first phoneme in the unknown pattern provides an index to reference patterns starting with that particular phoneme (i.e. direct access to a subset of candidates most likely to match). This method relies on the premise that the first phoneme in the unknown can be identified accurately. Another assumption is that as hypothesized portions are parsed from the unknown sequence, accurate word boundaries are realized. However, due to probable insertion, deletion, and substitution errors, the locations of word boundaries are not certain. (2) Organize the reference patterns based on broad phonetic classification, providing fast access to a restricted set of word candidates. Studies by Shipman et al. (1982), and Huttenlocher et al. (1984), found that by mapping phonemes of words into broad categories (vowels, stops, glides, and others), one-third of a 20,000 word lexicon could be uniquely specified with an approximate maximum of 200 words per category.

Sakoe and Chiba (1978) detail five general conditions that typically restrict the warping function. The first two are that the function be monotonic and continuous. Phonemes in the reference and unknown patterns are assumed to be time ordered with their intervals relatively uniform, satisfying the first two conditions. The three remaining conditions (boundary, adjustment window, and slope constraint) are variable and can affect the relative performance of the warping procedure. These three conditions will be held constant until an initial evaluation is performed on the most basic DTW algorithm and search procedures.

Boundary conditions (i.e., sequence endpoints) are fully known for both the reference and unknown patterns in isolated word recognition. However in continuous speech, endpoints (at the word level) in the unknown utterance are not fully established, and can be highly variable in number and position. Mapping a single reference word to a disproportionately long phrase would produce an unrealistic correspondence, in addition to wasting computational resources. Therefore, some criteria must be established for selecting the appropriate length of the unknown sequence for DTW comparison. Assuming the front-end's error rate is below 100 percent, one can hypothesize that there is a maximum number of phonemes (including errors) in the unknown pattern which must be examined in order to find a word, or exhaust all possibilities. This value would be equal to the phoneme count of the reference word, plus a buffer to allow for insertion errors that can extend the unknown sequence. Initially, this buffer value will be the same number of phonemes as the reference pattern. As in Picone's (1986) Text Alignment, a non-phoneme character is prepended and appended to both the reference pattern and some maximum length portion of the unknown sequence to be examined. The above results in the establishment of endpoints, satisfying the boundary condition for warping.

The adjustment window and slope constraint conditions affect the degree in which the DTW procedure accepts insertion and deletion errors. When finding a least cost path through the distance matrix, the warping path will cut a diagonal line with a slope of one if both patterns are time aligned. Deviation from the diagonal indicates larger differences between the two patterns. Excessively long horizontal or vertical paths indicate that unusual expansion or compression is required to match two patterns. The adjustment window forms a diagonal corridor somewhat parallel to the warping function. This "window" constant has the effect of limiting the number of acceptable insertion and deletion errors.

Kruskal and Sankoff (1983), Myers et al.(1981), and Sakoe et al. (1978) demonstrated the use of slope constraints which defined a parallelogram surrounding an optimal (diagonal) warping path. Calculations that result in a path that extends beyond this boundary are terminated, constraining the number of insertion and deletion errors that would otherwise result in excessively long paths. It also precludes wasteful calculations. Sakoe and Chiba's (1978) study showed that optimum DTW performance was maximized at a slope value of 1 in a range from 0.5 to 2. This slope value, combined with weighting coefficients to penalize for insertion, deletion, and substitution errors, results in the following recurrence relation to be initially used for the DTW procedure:

$$D(ai,bj) = \min \begin{cases} d(ai-1,bj-2)+w(sub/mat)+d(ai,bj-1)+w(del)+d(ai,bj) \\ d(ai-1,bj-1)+w(sub/mat) \\ d(ai-2,bj-1)+w(sub/mat)+d(ai-1,bj)+w(ins)+d(ai,bj) \end{cases}$$

where:

w(sub/mat) = penalty of either a substitution or a match

w(del) = penalty of a deletion error

w(ins) = penalty of an irsertion error

Averaging the total accumulated distance over the reference pattern length can be used to normalize distances between hypotheses whose reference patterns differ in length. This also favors a heuristic that looks for the longest pattern with minimum distance. All the conditions and variables above can be

adjusted to optimize the recognition procedure. But until an initial evaluation of a basic algorithm is done, no optimization will be attempted.

As previously mentioned , the system front-end is not complete and requires that we simulate it through the use of human confusability and vowel classification studies. Test data and error conditions must be simulated based on those studies as well. It is hoped that by observing algorithm performance as a function of different error conditions and variable settings, one can then predict the performance to be expected after the system's front-end response characteristics are actually determined.

Since many factors (as described above) influence the performance of the DTW algorithm, our initial approach will be to use a simple, brute force search technique, comparing all lexical entries to error-free unknown sequence portions. This basic approach will provide a performance benchmark from which to make comparisons as adjustments are made to system variables. Note that when comparing an error-free unknown to a reference pattern, any increase over zero in the accumulated distance indicates a difference in alignment, a basis for rejecting the reference. This establishes an initial threshold for testing purposes. As the threshold for rejection is raised, it's expected that beyond some limit, the ratio of incorrect hypotheses to correct ones will increase. Though this threshold is of primary importance in determining which hypotheses are accepted and which are rejected, we have no estimate of what its optimum value might be.

Other necessary tests will include examining how changes in the percentage of substitution, deletion, and insertion errors affect the ratio of incorrect to correct hypotheses. Initially the three error types will be assumed independent, and tested separately. It is not clear how, or if these errors are related. These tests can be executed with the distance threshold value constant, or by examining its effect at various levels. Changing the placement of errors may have a significant effect. Therefore, additional tests will compare the DTW performance when errors are evenly dispersed, as opposed to concentrated in consecutive series. Once a phoneme has been selected for substitution, a determination must be made as to what phoneme it will be exchanged with. Similarly, once a location has been determined for an insertion or deletion error, one must select the most appropriate phoneme (given its surrounding environment) to be inserted or deleted. A procedure to do this is under development and will be based primarily on using phonemes that are highly confusable.

6.6.4 Future Directions

As the DTW implementation is in its initial stages, it is difficult to predict what level of performance will be achieved. Once more is learned about the effects of various constraints and variables, efforts to optimize can take place. Most likely this will involve the development of methods to reduce the word candidate search space. Of course, once the front-end of the system is available, then more appropriate confusability data can be established for the distance measure. In addition, with a fully operational front end, additional probabilistic data can be established allowing the construction of Markov Models. This will provide an opportunity to compare each method and their respective advantages.

Bibliography

Baker, J.K. "The DRAGON System - An Overview," IEEE Transactions on Acoustics, Speech, and Signal Processing, 23, 1975, 24-29.

Cohen, P.S. and Mercer, R.L. "The Phonological Component of an Automatic Speech-Recognition System," in Speech Recognition: Invited Papers Of The 1974 IEEE Symposium, Reddy, D.R. ed., Academic Press, New York, 1975, 275-320.

Hillenbrand, J. and Gayvert, R.T. "Speaker-Independent Vowel Classification Based on Fundamental Frequency and Format Frequencies", Journal of the Acoustical Society of America, Spring 1987, 81 (Suppl. 1), S93 (A).

Huttenlocher, D., and Zue, V. "A Model of Lexical Access from Partial Phonetic Information", Proceedings ICASSP 1984, CH1945- 5/84/0000-0277.

Itakura, F. "Minimum Prediction Residual Principle Applied to Speech Recognition", IEEE Transactions Acoustical, Speech, Signal Processing, ASSP-23, 1975, 66-72.

Klatt, D.H. "Word Verification in a Speech Understanding System", in Speech Recognition: Invited Papers Of The 1974 IEEE Symposium, Reddy, D.R. ed., Academic Press, New York, 1975, 321-341.

Kruskal, J.B. and Sankoff, D. "An Anthology of Algorithms and Concepts Fop-4 Sequence Comparison", in Time Warps, String Edits, and Macromolecules: The Theory And Practice Of Sequence Comparison, Sankoff, D. and Kruskal, J.B. eds., Addison-Wesley, Reading, Ma., 1983, 265-311.

Levinson, S.E., Rabiner, L.R., and Sondhi, M.M. "Speaker

Independent Isolated Digit Recognition Using Hidden Markov Models", Proceedings of ICASSP Boston, 3, 1983, 1049-1052.

Levinson, S.E. " Continuous Speech Recognition by Means of Acoustic/Phonetic Classification Obtained from a Hidden Markov Model", Proceedings of ICASSP Dallas, 1, 1987, 93-95.

Lizza, Capt.G., Munger, M., Small, Capt.R., Feitshans, G., and Detro, S. "A Cockpit Natural Language Study - Data Collection and Initial Data Analysis," Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Ohio, April 1987, Doc. AFWL-TR-86-3003.

Lowerre, B. and Reddy, D.R. "The Harpy Speech Understanding System", in Trends In Speech Recognition, Lea, W.A. ed., Prentice Hall, Englewood Cliffs, N.J., 1980, 101-124.

Miller, G. and Nicely, P. "An Analysis of Perceptual Confusions Among Some English Consonants", Journal of the Acoustic Society of America, 27, 1955, 338-352.

Myers, C.S. and Rabiner, L.R. "A Level Building Dynamic Time Warping Algorithm for Connected Word Recognition," IEEE Transactions on Acoustics, Speech, and Signal Processing, 29, 1981, 286-297.

Ney, H. "The Use of a One-Stage Dynamic Programming Algorithm for Connected Word Recognition," IEEE Transactions on Acoustics, Speech, and Signal Processing, 32, 1984, 263-271.

Oskika, B.T., Zue, V.W., Weeks, R., Neu, H., and Aurbach J. "The Role of Phonological Rules in Speech Understanding Research", IEEE Trans. on Acoustics, Speech and Signal Processing, ASSP-23, No. 1, 1975, 104-112.

Picone, J., Goudie-Marshall, K., Doddington, G. and Fisher, W. "Automatic Text Alignment for Speech System Evaluation", IEEE Transactions on Acoustics, Speech, and Signal Processing, 34, 1986, 780-784.

Rabiner, L.R. and Juang, B.H. "An Introduction to Hidden Markov Models", IEEE ASSP Magazine, January 1986, 4-16.

Rudnicky, A., Baumeister, L., DeGraaf, K., and Lehmann, E. "The Lexical Access Component of the CMU Continuous System", Proceedings ICASSP Dallas, 1, 1987, 376-379.

Sakoe, H. and Chiba, S. "Dynamic Programming Algorithm Optimization for Spoken Word Recognition", IEEE Transactions on Acoustics, Speech, and Signal Processing, 26, 1978, 43-49.

Sakoe, H. "Two-Level DP-Matching -- A Dynamic Programming-Based Pattern Matching Algorithm for Connected Word Recognition", IEEE Transactions on Acoustics, Speech, and Signal Processing, 27, 1979, 588-595.

Shepard, R. "Multidimensional Scaling, Tree-Fitting, and Clustering", Science, 210, October 1980, 390-398.

Shipman, D.W. and Zue, V.W. "Properties of Large Lexicons: Implications for Advanced Isolated Word Recognition Systems", ICASSP 1982 Proceedings, 1982, 546-549.

Smith, A.R. and Sambur, M.R. "Hypothesizing and Verifying Words for Speech Recognition", in Trends In Speech Recognition, Lea, W.A. ed., Prentice Hall, Englewood Cliffs, N.J., 1980, 101-124.

Waibel, A. and Yegnanarayana, B. "Comparative Study of Nonlinear Time Warping Techniques in Isolated Word Speech Recognition Systems", Research paper, Carnegie-Mellon University, CMU-CS-81-125, June 1981.

Watari, M. "New DP Matching Algorithms for Connected Word Recognition", Proceedings of ICASSP Tokyo, 2, 1986, 1113-1116.

Wolf, J.J. and Woods, W.A. "The HWIM Speech Understanding System", IEEE Internat. Conf. Record on Acoustics, Speech, and Signal Processing, May 1977, 784-787.

Woods, W.A. "Motivation and Overview of SPEECHLIS: An Experimental Prototype for Speech Understanding Research", IEEE Trans. Acoustics, Speech, and Signal Processing, ASSP-23, 1975, 2-10.